PCT WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 6 : G06F 15/16, 13/42

A1

(11) International Publication Number: **WO 98/57268**

(43) International Publication Date: 17 December 1998 (17.12.98)

(21) International Application Number: PCT/US98/11801

(22) International Filing Date: 9 June 1998 (09.06.98)

(30) Priority Data:
08/871,704 9 June 1997 (09.06.97) US

(71) Applicant: MCI COMMUNICATIONS CORPORATION [US/US]; 1133 19th Street, N.W., Washington, DC 20036 (US).

(72) Inventors: SWIFT, Kim, Maurice; 1881 Valley View Court, Woodland Park, CO 80863 (US). BARRY, Edward; 6575 Denim Drive, Colorado Springs, CO 80918 (US). KETTLE, Bruce, Robert; 334 Winding Meadow Way, Monument, CO 80132 (US). LYNCH, David, Alan; 142 Yale Avenue, Colorado Springs, CO 80904 (US). SPIEKER, Jonathan, Lyle; 6265 Catskill Lane, Colorado Springs, CO 80918 (US).

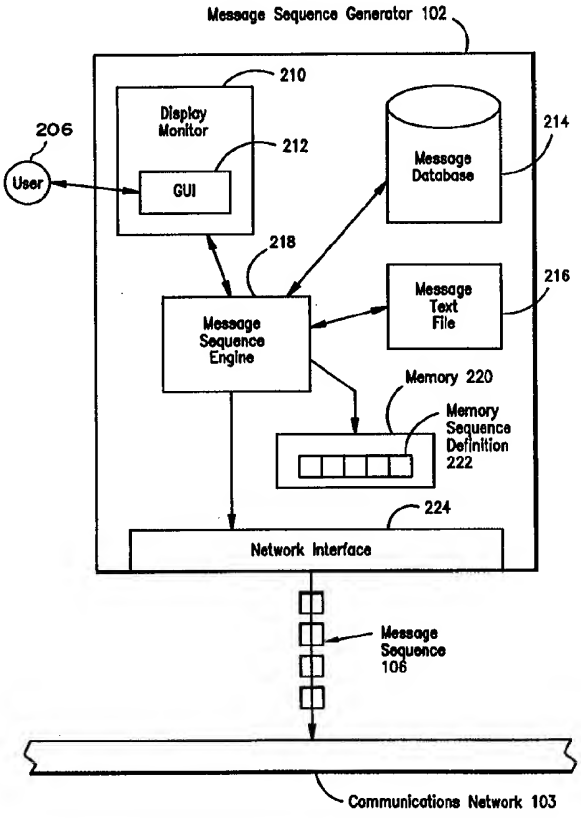
(81) Designated States: AU, CA, JP, MX, European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE).

Published
With international search report.
Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.

(54) Title: MULTI–PROTOCOL MESSAGE SEQUENCE GENERATOR

(57) Abstract

A system and method enabling a user to define a sequence of messages and transmit the messages to a target network object for testing. The use of the invention requires no generation, development, or modification of source code. A graphical user interface (GUI 212) is provided that allows a user to simply select the message type, content, and sequencing the user wishes to generate. The GUI also allows a user to select actual messages (212) that had been generated by actual network source objects. These messages can be sent to network management systems for testing of actual network events. The present invention also allows a user to edit existing messages. These functions provide a user with a wide array of options for creating virtually any testing scenario in a quick, easy, and efficient manner.



FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	**ES**	Spain	**LS**	Lesotho	**SI**	Slovenia
AM	Armenia	**FI**	Finland	**LT**	Lithuania	**SK**	Slovakia
AT	Austria	**FR**	France	**LU**	Luxembourg	**SN**	Senegal
AU	Australia	**GA**	Gabon	**LV**	Latvia	**SZ**	Swaziland
AZ	Azerbaijan	**GB**	United Kingdom	**MC**	Monaco	**TD**	Chad
BA	Bosnia and Herzegovina	**GE**	Georgia	**MD**	Republic of Moldova	**TG**	Togo
BB	Barbados	**GH**	Ghana	**MG**	Madagascar	**TJ**	Tajikistan
BE	Belgium	**GN**	Guinea	**MK**	The former Yugoslav Republic of Macedonia	**TM**	Turkmenistan
BF	Burkina Faso	**GR**	Greece			**TR**	Turkey
BG	Bulgaria	**HU**	Hungary	**ML**	Mali	**TT**	Trinidad and Tobago
BJ	Benin	**IE**	Ireland	**MN**	Mongolia	**UA**	Ukraine
BR	Brazil	**IL**	Israel	**MR**	Mauritania	**UG**	Uganda
BY	Belarus	**IS**	Iceland	**MW**	Malawi	**US**	United States of America
CA	Canada	**IT**	Italy	**MX**	Mexico	**UZ**	Uzbekistan
CF	Central African Republic	**JP**	Japan	**NE**	Niger	**VN**	Viet Nam
CG	Congo	**KE**	Kenya	**NL**	Netherlands	**YU**	Yugoslavia
CH	Switzerland	**KG**	Kyrgyzstan	**NO**	Norway	**ZW**	Zimbabwe
CI	Côte d'Ivoire	**KP**	Democratic People's Republic of Korea	**NZ**	New Zealand		
CM	Cameroon			**PL**	Poland		
CN	China	**KR**	Republic of Korea	**PT**	Portugal		
CU	Cuba	**KZ**	Kazakstan	**RO**	Romania		
CZ	Czech Republic	**LC**	Saint Lucia	**RU**	Russian Federation		
DE	Germany	**LI**	Liechtenstein	**SD**	Sudan		
DK	Denmark	**LK**	Sri Lanka	**SE**	Sweden		
EE	Estonia	**LR**	Liberia	**SG**	Singapore		

Multi-Protocol Message Sequence Generator

Background of the Invention

Field of the Invention

The present invention relates to systems and methods for testing, in a controlled manner, the response of network objects to the reception of messages from other network objects and, in particular, to systems and methods for conveniently generating and transmitting multi-protocol message sequences from a general purpose computer to a network object under test, whereby the messages appear as if they were generated by actual network objects.

Related Art

The rapid growth in the number, size, and importance of computer networks has greatly increased the need to verify that components to be deployed within a network, such as network management systems, will preform exactly according to specification. Accordingly, there is a need to test the response of network management systems in a simulated network environment.

Network management systems receive events (messages) from a wide variety of network components, such as network switches and network routers. The management systems are programmed to respond in a specific way to certain events (messages) received from certain network components. For example, if a network switch sends a message to the management system indicating that the switch's capacity has been reached, the network management system is programmed to respond by setting off an alarm. An individual in charge of maintaining the network will respond to the alarm and, it is hoped, fix the capacity problem. In this manner, a network is efficiently monitored and all problems that arise can be quickly dealt with.

Prior to deploying a management system into a network, the management system's response to the reception of events (messages) must be thoroughly tested. This requires a

controlled testing environment in which messages or message sequences are sent to the management system to verify that it responds properly.

One solution for providing this controlled testing environment, in which messages or message sequences are sent to the management system, consists of creating customized software programs that emulate or simulate the behavior of certain network components. This approach, however, has significant drawbacks. First, to accurately emulate the components of a network all possible scenarios must be accounted for in the emulation program. It is virtually impossible, and at the very least, extremely expensive, to create an emulation that can account for the hundreds of possible scenarios.

Additionally, emulation programs once created are difficult to use and maintain. Each time a network component is changed the emulation program must also be modified to account for all the changes. Time consuming and effort intensive modifications to the emulation program would also be needed to support a change in the underlying communications protocols.

Another possible way to perform controlled tests of a network management system is to assemble actual network components in a laboratory and manually cause the components to send messages to the management system. With this approach, however, it is often impossible to obtain complete coverage of all testing scenarios because it is difficult to manually cause the components to send certain alarm messages.

What is needed is a reliable and easy to use application that enables a user to construct any desired sequence of messages and send them to a target network management system for testing. Furthermore, use of the application should not require the generation, development, or modification of computer program source code.

Summary of the Invention

The present invention provides a system and method for allowing a user to generate and transmit message sequences. The message sequences are transmitted to network target objects so that the user can verify the response of a network target object to the reception of a particular message sequence.

The system and method further provide a means for the user to create a message sequence definition for specifying the message sequence to be generated. Additionally, the system provides a convenient and easy to use graphical user interface for creating the message sequence definition. The system also includes storage means for saving and retrieving message sequence definitions and message sequence items that comprise the message sequence definition.

The present invention transmits message sequences from a message sequence generator to a data collector. The data collector stores the message sequence into a particular store forward file based on the network target object to receive the message sequence. A data distributor continuously monitors the store forward file. After the data collector has stored a message into the store forward file, the data distributor will read the file, create a copy of the message, apply, if necessary, a filter to the message based on the data distributor's configuration file, and transmit the message to the intended target object. Alternatively, the data distributor could be configured to apply a filter to each message based on the network source object associated with each message, instead of based on the configuration file.

In another alternative embodiment of the invention, the message sequence generator sends each message of a message sequence to a particular data collector based on the network source object type associated with the message. Thus, only messages that share the same network source object type will be transmitted to a particular data collector. Each data collector will store the messages in a particular store forward file based on the network target object. Each of the store forward files will be monitored by a single data distributor. Each data distributor will read in messages from the file it is monitoring, apply a filter to the

messages based on a configuration file, and transmit the messages to the intended target object.

In yet another alternative embodiment of the invention, the message sequence generator transmits all message sequences to a single data collector. The data collector will store the individual messages in a particular store forward file based on the network source object type and network target object associated with the message. Each store forward file will be monitored by a data distributor. The data distributor will read in messages from the file it is monitoring, apply a filter to the messages based on a configuration file, and transmit the messages to the intended target object.

In yet another embodiment, the message sequence generator transmits message sequences directly to target objects, thereby bypassing the data collector and data distributor.

Another object of the invention is to allow time slices of actual network messages to be transmitted to a target object. A "time slice" of messages is a set of network source object messages stored in a production store forward file that were generated within a certain period of time. Each message in a production store forward file has an associated time stamp indicating the time of day the message was generated. In this scenario, a data collector has access to copies of production store forward files, and the message sequence generator has the ability to transmit a command message to the data distributor. The command message supplies a start and stop time and the name of a file containing a copy of the contents of a production store forward file. The data distributor receives the command message from the message sequence generator and then reads in messages from the file that have time stamps that fall within the start and stop time specified by the command message. Those messages that were read in would then be filtered and transmitted to a target object. In this manner, time slices of data are transmitted to the target object.

The present invention enables users to construct any desired sequence of messages and send them to a target network management system for testing. Furthermore, use of the

present invention does not require the generation, development, or modification of computer program source code.

Summary of the Drawings

The foregoing and other features and advantages of the present invention will be apparent from the following, more particular description of a preferred embodiment of the invention, as illustrated in the accompanying drawings.

FIG. 1 is a high-level diagram illustrating the preferred controlled test environment.

FIG. 2 is a high-level diagram illustrating the preferred embodiment of the message sequence generator.

FIG. 3 illustrates a message sequence definition.

FIGS. 4A and 4B illustrates the graphical user interface used by the present invention with which the user interacts with to create a message sequence definition.

FIGS. 5A, 5B, and 5C are flow charts illustrating the operational flow of the message sequence generator in accordance with the present invention.

FIG. 6 is a flow chart illustrating the operational flow of the data collector in accordance with the present invention.

FIG. 7 is a flow chart illustrating the operation flow of the data distributor in accordance with the present invention.

FIG. 8 is a diagram illustrating a production network environment that the present invention is capable of simulating.

FIG. 9 is a diagram illustrating a test environment where time slices of data that came from actual network source objects can be transmitted to the network target object.

FIG. 10 is a high-level diagram illustrating an alternative test environment.

The preferred embodiment of the invention is now described with reference to the figures where like reference numbers indicate like elements. Also in the figures, the left most digits of each reference number correspond to the figure in which the reference number is first used.

Detailed Description of the Preferred Embodiment

The present invention provides a controlled test environment for verifying the response of target objects to the reception of message sequences. More specifically, the test environment includes a message sequence generator for defining and generating message sequences. The message sequences defined and generated by the message sequence generator correspond to actual message sequences transmitted by network source objects to target objects in a production network. By simulating the production network in this manner, the system and method of the present invention provide an accurate means for verifying the response of network target objects to the reception of message sequences in a controlled test environment.

The term network source object is a generic term referring to any device capable of communicating messages on a network. Such devices include but are not limited to: switches, routers, bridges, repeaters, etc. . . The term target network object is also a generic term. The term target network object refers to any network object that is responsive to the reception of a message sequence. Such network objects include but are not limited to network management systems.

FIG. 1 illustrates the preferred controlled test environment. The controlled test environment includes one or more message sequence generators 102 connected to a TCP/IP communications network 103, a data collector 108 also connected to the TCP/IP communications network 103, one or more store forward files 110 that are accessible by data collector 108, one or more data distributors 112 that monitor store forward files 110, and one or more target objects 114 that receive a message sequence 106 from data distributors 112. Data collector 108 and data distributor 112 are software applications which run on, for example, a DEC Alpha computer. Message sequence generator 102 will be described in more detail with reference to FIG. 2.

FIG. 2 illustrates the preferred embodiment of message sequence generator 102. The features of message sequence generator 102 include a message sequence engine 218, a

message database 214, a message text file 216, a display monitor 210, a graphical user interface (GUI) 212, a memory storage area 220 for storing message sequence definition 222, and a network interface 224. Network interface 224 is connected to communications network 103. In the preferred embodiment, communication network 103 is a Transport Control Protocol/Internet Protocol (TCP/IP) communication network, or any other communication network that can transfer the messages described herein.

Message sequence engine 218 is a software application running on a Personal Computer (PC) controlled by the Windows operating system. In the preferred embodiment, message sequence engine 218 is implemented using the PowerBuilder development environment with a PowerSockets library included. PowerBuilder is a fourth generation language that enables a programmer to develop windows graphically. PowerSockets is a library that can be used with PowerBuilder to produce a TCP/IP capable application. The invention, however, is not limited to the PowerBuilder development environment. Other development environments, such as the C or C++ development environments, may be used to implement the functionality of message sequence engine 218.

Message sequence engine 218, among other things, allows a user to create a message sequence definition 222 via GUI 212, and upon the user's request, causes network interface 224 to transmit message sequence 106, which corresponds to message sequence definition 222, onto communications network 103. Message sequence definition 222 will be described in more detail with reference to FIG. 3.

Message database 214 is a storage device that is accessible by message sequence engine 218. Message database 214 contains actual messages that have been sent by network source objects to target objects and/or user created messages. Message database 214 is populated with actual messages that had been sent from network source objects to target objects by importing the contents of production store forward files 810. Message database 214 also contains message sequence definitions that a user has previously created and saved to message database 214. Message text file 216 is similar to message database 214. Message text file 216 may contain actual messages that have been sent by network source objects to

target objects and/or user created messages, and it is accessible to message sequence engine 218. However, unlike message database 214, message text file 216 can be created by a user using a standard word processing application.

FIG. 3 illustrates message sequence definition 222. Message sequence definition 222 defines message sequence 106. Message sequence definition is stored in memory storage area 220 by message sequence engine 218. Message sequence definition 222 is created by a user interacting with message sequence engine 218 via GUI 212 and display 210. This process will be described in more detail with reference to FIGS. 4A and 4B.

Message sequence definition 222 comprises one or more sequence items 302A, 302B, . . ., 302N. Sequence item 302C is a representative sequence item. Sequence item 302C comprises four data records: (1) network source object identifier 304; (2) data message 306 to be sent to the target object; (3) the number of times to send the message 308; and (4) the delay time 310 between the sending of each data message 306. The number of times to send the message data record 308 may contain a value of 1.

FIGS. 4A and 4B shows GUI windows 404 (Message Generation window) and 430 (Edit Sequence Item window) with which the user interacts with to create message sequence definition 222. A user interacts with GUI windows 404 and 430 by means of a keyboard and a mouse with a left and a right button or similar pointing device (the keyboard and mouse are not shown). This type of point and click technology is well known in the art. Described below are the preferred steps a user would take in operating message sequence generator 102 via GUI 212 to create message sequence definition 222.

When a user wishes to test a target system by sending messages to the target, the user will invoke message sequence engine 218. Upon invocation, the message sequence engine will display Message Generation window 404. When Message Generation window 404 is first displayed, message sequence definition 222 does not contain any sequence items.

Message Generation window 404 includes a message sequence text entry field 406, a target system pull down window 408, a message sequence display window 410, a pop-up menu 412 which is hidden from the user until the user presses the right mouse button while the mouse pointer is within the boundaries of message sequence display window 410, a Load button 414, a Save button 416, a Generate button 418, a Clear button 420, and a Cancel button 422. Message sequence display window 410 displays all sequence items 302 of message sequence definition 222. At start up, message sequence data window 410 is clear because no sequence items 302 have been added to message sequence definition 222.

There are two ways for a user to add sequence items to the empty message sequence definition 222. First, a user could input a sequence name in message sequence text entry field 406 and then click on Load button 414. Clicking on the Load button signals message sequence engine 218 to search message database 214 for a previously saved sequence definition having the name that the user specified in text entry field 406. If one is found, message sequence engine 218 will load from message database 214 the corresponding sequence items into message sequence definition 222.

The second way of adding sequence items to the empty message sequence definition is to manually add them to the definition. Pressing the right mouse button in message sequence display window 410 creates a pop-up menu 412 on the display. Pop-up menu 412 enables a user to add, insert, change, copy, delete, or restore a sequence item. To add a sequence item 302 to the message sequence definition 222 a user would select the "Add" pop-up menu option. Upon selecting the "Add" option, message sequence engine 218 will display Edit Sequence Item window 430 (FIG. 4B) and add a blank sequence item 302 to the sequence definition 222.

Edit Sequence Item window 430 includes: a message text window 432; a menu bar 434; a Device Name drop-down window 436; a Specific Device drop-down window 438; input fields 440, 442, 444, 446, 448, and 450 for specifying the number of times to generate the message and the delay after each message respectively; a Paste toggle button 452; an Add button 454; an Insert button 456; a Change button 458; and a Cancel button 460.

When Edit Sequence Item window 430 is first displayed, Paste toggle button 452 is not activated and Add, Insert, and Change buttons 454-458 are inactive. When the Past toggle button is inactivated, any changes to the sequence item 302 in the Edit Sequence Item window 430 will automatically appear in message sequence display window 410, and will automatically become part of message sequence definition 222. However, when Paste toggle button is activated the buttons Add, Insert and Change become active and any changes to the sequence item will not automatically appear in message sequence display window 410. To have changes in the sequence item appear in the message sequence data window while in Paste mode, a user must select either the Add, Insert, or Change button.

Edit Sequence Item window 430 enables a user to specify all of the necessary elements of a sequence item 302 (see FIG. 3). For example, the device name drop-down data window 436 allows a user to specify the network source object 304. To specify data message 306 that is to be transmitted to one of the target objects 114, a user has two options. First, a user could simply use the keyboard to type a data message into message text window 432. Alternatively, a user could select the "Open Message" menu option from menu bar 434. Selecting this option presents a user with four additional options. These options are: (1) Open Formatted Message Catalog; (2) Open Formatted Message File; (3) Open Unformatted Message Catalog; and (4) Open Unformatted Message File. Selecting options (1) or (3) will present a user with formatted or unformatted messages that are stored in message database 214. Selecting options (2) or (4) will present a user with formatted or unformatted messages that are stored in message text file 216.

A formatted message, as opposed to an unformatted message, is a message that has been processed by a filter. A filtering process is a process that modifies a message so that the message can be understood by a target system. The filtering process can be thought of as a translation process. An example filter could be one that adds a header to each message from a particular network source object. A reason filtering may be required is that certain source objects and target objects may have been manufactured by different vendors and thus can not communicate directly without message filtration. The present invention, therefore, gives a user flexibility in that the user can select either pre-filtered messages (i.e. formatted messages)

or non-filtered messages (i.e. unformatted messages). If the user selects unformatted messages then the message filtering may be performed by data distributor 112, depending on its configuration file, as will be discussed in more detail with reference to FIG. 7.

After the messages are presented to a user, the user simply clicks on the message he or she desires to include in the sequence item. The message will then be displayed in message text window 432. The message text window 432 provides a means for the user to edit data messages.

Upon completely specifying the sequence item 302 through Edit Sequence Item window 430, a user can simply close the window to get back to the main Message Generation window 404 where the sequence item 302 will be displayed in message sequence display window 410. At this point a user can add additional sequence items 302 to message sequence definition 222 or insert, change, copy, delete, or restore an existing sequence item. A user can perform these functions by once again positioning the mouse in message sequence data window 410 and clicking on the right mouse button which will bring up pop-up menu 412.

Before the messages defined in message sequence definition 222 are transmitted to a target object, a user must specify the target object that is to receive the messages. Target system drop-down data window 408 enables the user to perform this step. Upon creating the sequence definition and specifying the target object to receive the messages, a user may send a generate signal to message sequence engine 218 by clicking on the Generate button 418. The generate signal signals the message sequence engine 218 to begin the process of transmitting message sequence 106, which corresponds to message sequence definition 222, to a target object. A user may also click on Save button 416 which will cause the message sequence definition to be stored in the message database 214 for subsequent retrieval.

FIGS. 5A, 5B, and 5C illustrate a flow chart 500 depicting the operation of message sequence engine 218. The flowchart 500 is of sufficient detail to enable one skilled in the relevant art to generate a computer program or a computer program product in accordance with the present invention. A user 206 wanting to test the response of a target object to the

reception of a message sequence will invoke message sequence engine 218. As stated earlier, message sequence engine 218 is a Windows application and it is well known in the art how to invoke it. Upon invocation, message sequence engine 218 will create an empty message sequence definition 222 and display Message Generation window 404. This is shown in steps 504 and 506, respectively. Next, message sequence engine 218 will wait for user 206 to take action. This is shown in step 508. Message sequence engine 218 will react in a specific way to a specific user action, therefore, it will determine which action the user has taken. This is shown in steps 510, 524, 530, 560, 568, and 576.

If user 206 activates Load button 414, message sequence engine 218 will retrieve a message sequence identifier from message sequence text entry field 406. This is shown in step 512. Message sequence engine 218 will then search message database 214 for a previously saved sequence definition having the same identifier as the one specified in message sequence text entry field 406. This is shown in step 514. In step 516, message sequence engine 218 will determine if it found what it was searching for. If a previously saved sequence definition is found, then message sequence engine 218 will load from message database 214 the corresponding sequence items into message sequence definition 222 (as shown in step 520) and control will pass to step 522, else an error message will be displayed (step 518) and control will pass back to step 508. In step 522, message sequence engine 218 will display all of the sequence items 302 of message sequence definition 222 in message sequence display window 410. After step 522, control passes back to step 508 where message sequence engine 218 will wait for another user action.

If user 206 activates Save button 416, message sequence engine will retrieve a message sequence identifier from message sequence text entry field 406. This is shown in step 526. Next, engine 218 will store sequence items 320, which comprise the message sequence definition 222, into message database 214 together with the identifier so that the sequence items can be retrieved at a later time. This is shown in step 528. After completing step 528, control returns to step 508 where message sequence engine 218 will wait for another user action.

If user 206 activates the Generate button 418, message sequence engine 218 will retrieve a target system identifier from target system pull down window 408. This is shown in step 532. Next, it will determine whether there is an existing TCP/IP connection with data collector 108. This is shown in step 534. If there is not an existing TCP/IP connection, control will proceed to step 540, otherwise message sequence engine will determine whether the target system identifier has changed since the previous generate request (step 536). If the identifier has changed, control passes to step 538, otherwise control passes to step 544.

In step 538, the existing TCP/IP connection is closed by message sequence engine 218. After closing the connection, message sequence engine directs network interface 224 to transmit a TCP/IP connect request to data collector 108 (step 540). Once the connection is established, registration message 104, which contains, among other items, the target system identifier, is transmitted to data collector 108 (step 542). Sending registration message 104 establishes a session between message sequence engine 218 and data collector 108. The target system identifier contained in the registration message 104 identifies target object 114A-D that is to receive message sequence 106. The target system identifier is used by data collector 108, as will be discussed with reference to FIG. 6.

In step 544, it is determined whether there is at least one sequence item in message sequence definition 222. If there isn't, then an error message is generated (step 558) and control passes to step 508, otherwise control passes to step 546. In step 546, the first sequence item is retrieved from message sequence definition 222. Next, the number of times to send data message 308 and the delay value 310 are determined from the contents of the sequence item (steps 548 and 550). Following this, message sequence engine 218, utilizing network interface 224, transmits data message 306 the number of times specified in the sequence item with a delay between each message equal to the delay specified in the sequence item (step 552). As an illustrative example, assume the sequence item contains the following data: data message 306 = "Hello World"; number of times to send message 308 = 3; delay time 310 = 5 seconds. Given this scenario, engine 218 will send a sequence of three messages to the data collector. The three messages will contain the data "Hello World" and there will be

a 5 second delay between the transmission of each message. After step 552, control passes to step 554.

In step 554, it is determined whether there are more sequence items to process. If there are no more to process control passes to step 508, otherwise control passes to step 556. In step 556 the next sequence item from the message sequence definition 222 is retrieved . Control then passes back to step 548. In this manner, all sequence items in message sequence definition 222 will be processed.

If user 206 activates Clear button 420, message sequence engine 218 will ask user 206 for confirmation, as is shown in step 562. Upon receiving confirmation, message sequence engine will clear message generation window 404 and re-initialize message sequence definition 222 such that it will not contain any sequence items. This is shown in steps 564 and 567 respectively. If confirmation is not received, control passes to step 508.

Similarly, if user 206 activates Cancel button 422, message sequence engine will ask user 206 for confirmation, as shown in step 570. Upon receiving confirmation, message sequence engine will close the message generation window 404 and terminate itself. This is shown in step 574.

Lastly, if user 206 activates the pop-up menu by pressing on the right mouse button while the mouse pointer is within the message sequence display window 410, message sequence engine 218 will display pop-up menu 412. This is shown in step 578. Next, engine 218 will get the user's selection. If user 206 selected Add from the pop-up menu, message sequence engine 218 will display a blank Edit Sequence Item window 430. This is shown in step 586. In step 588, message sequence engine 218 will receive input from user 206 specifying a sequence item. How a user interacts with the Edit Sequence Item window has been described with reference to FIG. 4B. In step 590, message sequence engine will add the sequence item to message sequence definition 222 and display the sequence item in message sequence display window 410, as shown in step 592. After completing step 592, control passes back to step 508. If user 206 does not select Add from the pop-up menu, then engine

218 will process either the insert, change, delete, or restore selection, as shown in step 584, and then control will return to step 508. The details of how these selections are processed is not shown for it is well within the ability of one skilled in the art to determine how those selections should be processed in view of the description of how the Add selection is processed.

FIG. 6 illustrates a flow chart 602 depicting the operation of data collector 108. The flowchart 602 is of sufficient detail to enable one skilled in the relevant art to generate a computer program or a computer program product in accordance with the present invention. FIG. 6 starts with step 606 , where control immediately passes to step 610. In step 610, the data collector waits for an incoming TCP/IP connect request from the message sequence generator 102. If a connect request is received, data collector 108 will accept the request (step 612), which will create a logical connection between message sequence generator 102 and data collector 108. Data collector 108 then waits for message sequence generator 102 to transmit a registration message 104 (step 614). Once a registration message is received, data collector 108 will determine if registration message 104 is valid, as shown in step 618. If the registration message is not valid, step 622 is performed. In step 622, the data collector will close the TCP/IP connection, thereby logically disconnecting itself from message sequence generator 102. After closing the connection, data collector 108 will wait for the next incoming TCP/IP connection request, as shown in step 610.

If the registration message 104 is valid, then step 626 is performed. In step 626, the data collector 108 will use the target system identifier contained in registration message 104 to identify the store forward file 110 that will be used to store messages received on the established TCP/IP connection from message sequence generator 102. The target system identifier determines which store forward file 110 will be used because, in the preferred embodiment there is a one-to-one correspondence between store forward files 110 and target objects 114. In other words, in the preferred embodiment, for each target object 114, there is a single corresponding store forward file 110.

After step 626 is performed, control passes to step 630. In step 630, data collector 108 will wait for messages or a disconnect from message sequence generator 102. Upon receiving either a message or a disconnect control passes from step 630 to step 636. Step 636 is a decisional step. If a message was received control will pass to step 638, otherwise control passes back to step 622. In step 638, data collector 108 will store the message received in the identified store forward file. After completing this step, control goes back to step 630.

FIG. 7 illustrates a flow chart 702 depicting the operation of data distributor 112A. Data distributor 112A is a representative data distributor 112. The flowchart 702 is of sufficient detail to enable one skilled in the relevant art to generate a computer program or a computer program product in accordance with the present invention. FIG. 7 starts with step 704, where control immediately passes to step 706. In step 706, the data distributor will read its configuration file. The configuration file, among other things, controls whether data distributor 112A will filter messages and dictates what filter will used. After the completion of step 706, control passes to step 708. In step 708, the data distributor 112A retrieves the status of the store forward file 110A that it is monitoring. In the preferred embodiment, there is a one-to-one relationship between store forward files 110 and data distributors 112. If data distributor 112A determines that the status indicates that a new message has been stored in store forward file 110A (step 712), then control passes to step 716, otherwise control goes back to step 708. In step 716, data distributor 112A will read a portion of store forward file 110A and create a copy of the new message that was stored in the store forward file. Upon completion of this step, control passes to step 718. In step 718, data distributor 112A determines whether it should filter the copied message. If it is to apply a filter, then it will filter the copied message (step 720) and then transmit the filtered message to the target object (step 722). If it is not to apply a filter then it simply transmits the message to the appropriate target object (step 722). After step 722, control returns to step 708.

It should be noted that a data distributor 112 can be configured to use one of many communications protocols when communicating with a target object. As shown in FIG. 1, the four data distributors 112A-D are configured to communicate via TCP/IP, LU0, X.25, and

DECNET respectively. The invention, however, is not limited to these protocols or this number of data collectors and target objects.

The ability of the data distributors to communicate via a wide variety of communications protocols means that a single message sequence generator 102, which may be capable of only TCP/IP communication, is able to send messages to numerous target objects, regardless of the communication protocol a target object is configured to use. Consequently, a single message sequence generator 102 can be utilized in a wide variety of test environments without having to make any modifications to the message sequence generator 102.

Utilizing the controlled test environment as described above allows user 206 to verify the response of target object 114 to the reception of message sequence 106. Consequently, the response of target system 114 to a certain message sequence 106 can be verified in a convenient, flexible, controlled test environment.

FIG. 8 illustrates a typical production network 800. In production network 800, network source objects 802 communicate with target objects 814 using the same data collector/data distributor system of the test environment depicted in FIG. 1. Consequently, the test environment depicted in FIG. 1 accurately simulates the production environment, thus ensuring accurate test results.

Because the production network utilizes the data collector, there exist production store forward files 810 that contain actual network messages that were sent by source objects to target objects. It is possible to import the contents of production store forward files 810 into message database 214.

Additionally, the existence of the production store forward files allows message sequence generator 102 the ability to play back "time slices" of data that came from actual network source objects. A "time slice" of data is a set of network source object messages that occur within a certain period of time. Message sequence generator 102 is able to play back

time slices because all messages in the store forward files 810 have time stamps indicating the time at which the message was placed in the file.

FIG. 9 illustrates a test environment where time slices of data are transmitted to a target object 114. The test environment includes a message sequence generator 902, a data distributor 912, a TCP/IP communications network 903 for enabling the message sequence generator 902 to communicate with the data collector 912, one or more store forward files 910A-N, and a target object 914. Although only one message sequence generator, data distributor, and target object is illustrated, it should be appreciated by one skilled in the art that the test environment can include a plurality of message sequence generators, data distributors, and/or target objects.

The store forward files 910A-N are copies of production store forwards files 810A-N. A user (not shown) would direct message sequence generator 902 to send a command message 904 to data distributor 912. Command message 904 comprises the name of a store forward file and a start and stop time. Data distributor 912 receives command message 904 and proceeds to read messages from the named store forward file that have time stamps that fall between the start and stop time. Those messages 906 would then be transmitted to the target object. The data distributor may also filter the messages prior to transmitting them to the target object. Alternatively, Message sequence generator 902 could grow to include the retrieval of time slices of data from store forward files in the same way that Message sequence generator 902 has the ability to retrieve messages and message sequences from message database 214.

FIG. 10 illustrates another alternative test environment. As depicted in FIG 10, message sequence generator 102 can communicate directly with target systems via TCP/IP, or any other communication protocol that can transfer the messages described herein. Consequently, the data collector 108 and data distributor 112 are bypassed in the alternative test environment. The message sequence generator 102 in the alternative test environment operates exactly as it would in the test environment depicted in FIG. 1, except with respect to operations in response to user 206 activating the generate button. In the alternative

environment, only two steps of flowchart 500 need to be modified. First, step 540 should be changed from sending a TCP/IP connect request to data collector 108 to sending a TCP/IP connect request to the target object 114 that is identified by the target system identifier. Second, step 542 would not be performed in the alternative environment. Other than these two steps, message sequence generator 102 operates the same way in the preferred and the alternative test environments.

While various embodiments of the present invention have been described above, it should be understood that they have been presented by way of example, and not limitation. Thus the breadth and scope of the present invention should not be limited by any of the above-described exemplary embodiments, but should be defined only in accordance with the following claims and their equivalents.

What is claimed is:

1. A system for verifying the response of a network target object to the reception of a message sequence, comprising:

creating means for creating a message sequence definition comprising a sequence item; and

transmitting means for transmitting onto a communications network the message sequence, wherein the message sequence corresponds to said message sequence definition.

2. The system of claim 1, wherein said sequence item comprises:

a first data record representing a network source object identifier;

a second data record representing a data message;

a third data record representing a number of times to send the data message; and

a fourth data record representing a delay time.

3. The system of claim 1, wherein said creating means comprises:

a message sequence engine that controls the behavior of a graphical user interface (GUI), said graphical user interface (GUI) comprising a window for displaying said message sequence definition, a window for editing said sequence item, and an activatable button; and

a memory storage area accessible by said message sequence engine for storing said message sequence definition.

4. The system of claim 3, wherein said creating means further comprises a storage means, accessible by said message sequence engine, for storing and retrieving data messages and said message sequence definition.

5. The system of claim 4, wherein said activatable button allows the user to signal said message sequence engine to load from said storage means one or more sequence items into said message sequence definition.

6. The system of claim 1, wherein said transmitting means includes:

a message sequence engine; and

a network interface coupled to said message sequence engine;

said message sequence engine passing messages to said network interface and said network interface transmitting said messages onto said communications network.

7. The system of claim 1, further comprising:

a data collecting means for receiving a message transmitted by said transmitting means and for storing said received message in a store forward file;

a monitoring means for determining when said message has been stored in said store forward file;

a reading means for reading said message from said store forward file, said reading means being activated upon said monitoring means determining that said message has been stored in said store forward file; and

a distribution means for transmitting said message read by said reading means to the network target object.

8. The system of claim 7, further comprising:

a filtering means for filtering said message read by said reading means prior to said message being transmitted to the target object.

9. A method for verifying the response of a network target object to the reception of a message sequence, comprising the steps of:

creating a message sequence definition comprising one or more sequence items; and

transmitting onto a communications network the message sequence, wherein the message sequence corresponds to said message sequence definition.

10. The method of claim 9, wherein the step of creating a message sequence definition includes the step of using a graphical user interface to specify said one or more sequence items.

11. The method of claim 9, further comprising the steps of:

receiving a message transmitted by said transmitting means;

storing the message in a store forward file;

monitoring said store forward file to determine if the store forward file has been modified;

upon determining that the store forward file has been modified, reading the message from the store forward file; and

sending the message read from said store forward file to the network target object.

12. The method of claim 11, further comprising the step of filtering the message prior to sending the message to the network target object.

13. A system for generating a message sequence, comprising:

a message sequence engine;

a memory, coupled to said message sequence engine, for storing a message sequence definition; and

a network interface, coupled to said message sequence engine, for transmitting the message sequence, wherein the message sequence corresponds to said message sequence definition.

14. The system of claim 13, wherein said message sequence engine includes display means for displaying a graphical user interface, wherein a user interacts with said graphical user interface to specify sequence items to be included in said message sequence definition.

15. A system for testing a network target object, comprising:

a message sequence generator for transmitting a message onto a communications network;

a data collector for receiving said message from said communications network and storing said message in a store forward file; and

a data distributor for reading said message stored in said store forward file into a memory and transmitting said message stored in said memory to the network target object.

16. The system of claim 15, wherein said data distributor filters said message stored in said memory prior to transmitting said message to the network target object.

17. A system for testing a network target object, comprising:

a message sequence generator for transmitting a plurality of messages onto a communications network;

a data collector for receiving at least one of said messages from said communications network and storing said at least one of said messages in a store forward file; and

a data distributor for reading said messages stored in said store forward file into a memory and transmitting said messages stored in said memory to the network target object.

18. The system of claim 17, wherein said data distributor filters said messages stored in said memory prior to transmitting said messages to the network target object.

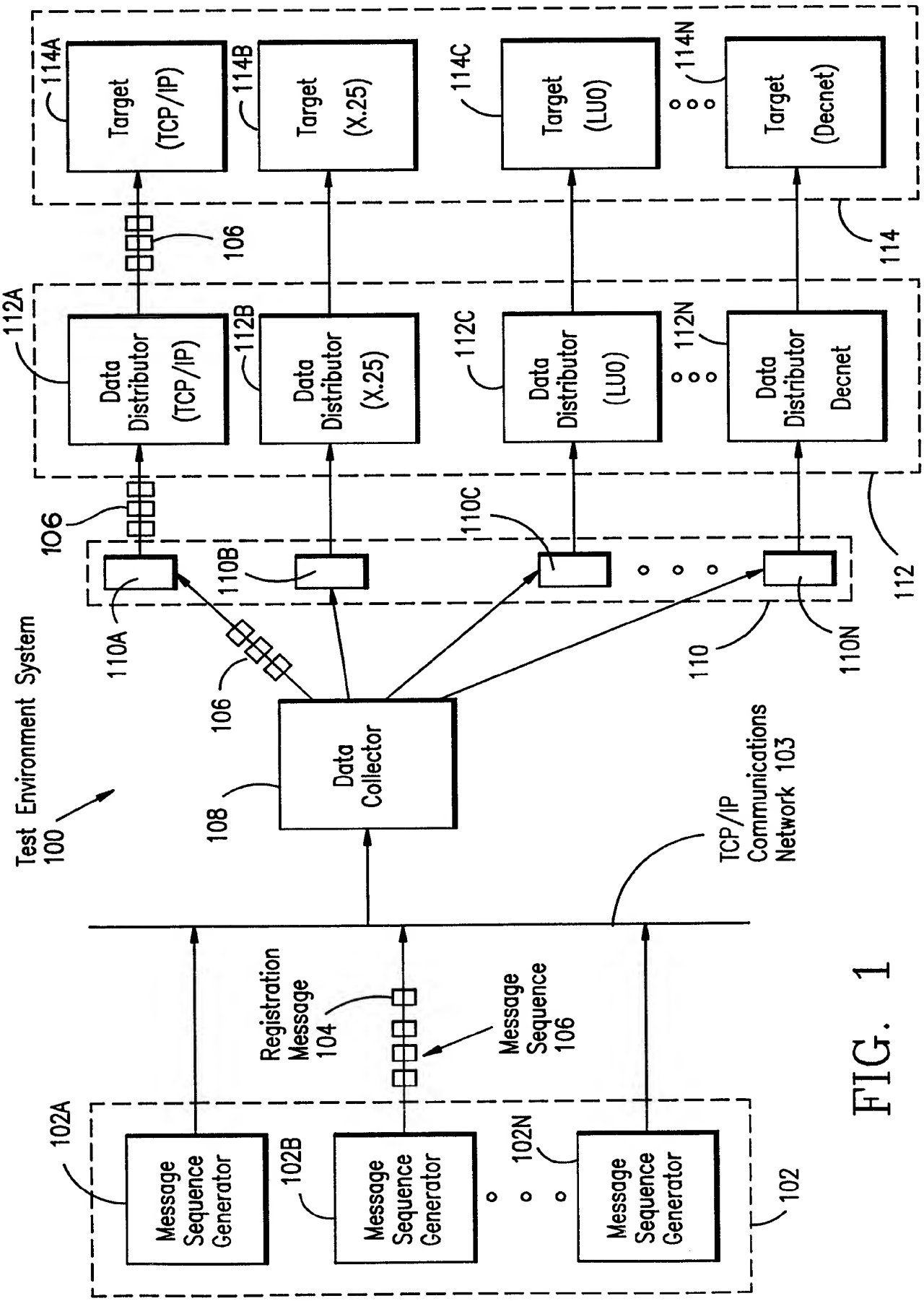

Test Environment System
100
Message Sequence Generator
102A
Message Sequence Generator
102B
Message Sequence Generator
102N
102
Registration Message 104
Message Sequence 106
TCP/IP Communications Network 103
Data Collector
108
106
110A
110B
110C
110
110N
112
Data Distributor (TCP/IP)
112A
Data Distributor (X.25)
112B
Data Distributor (LU0)
112C
Data Distributor Decnet
112N
106
114
Target (TCP/IP)
114A
Target (X.25)
114B
Target (LU0)
114C
Target (Decnet)
114N

FIG. 1

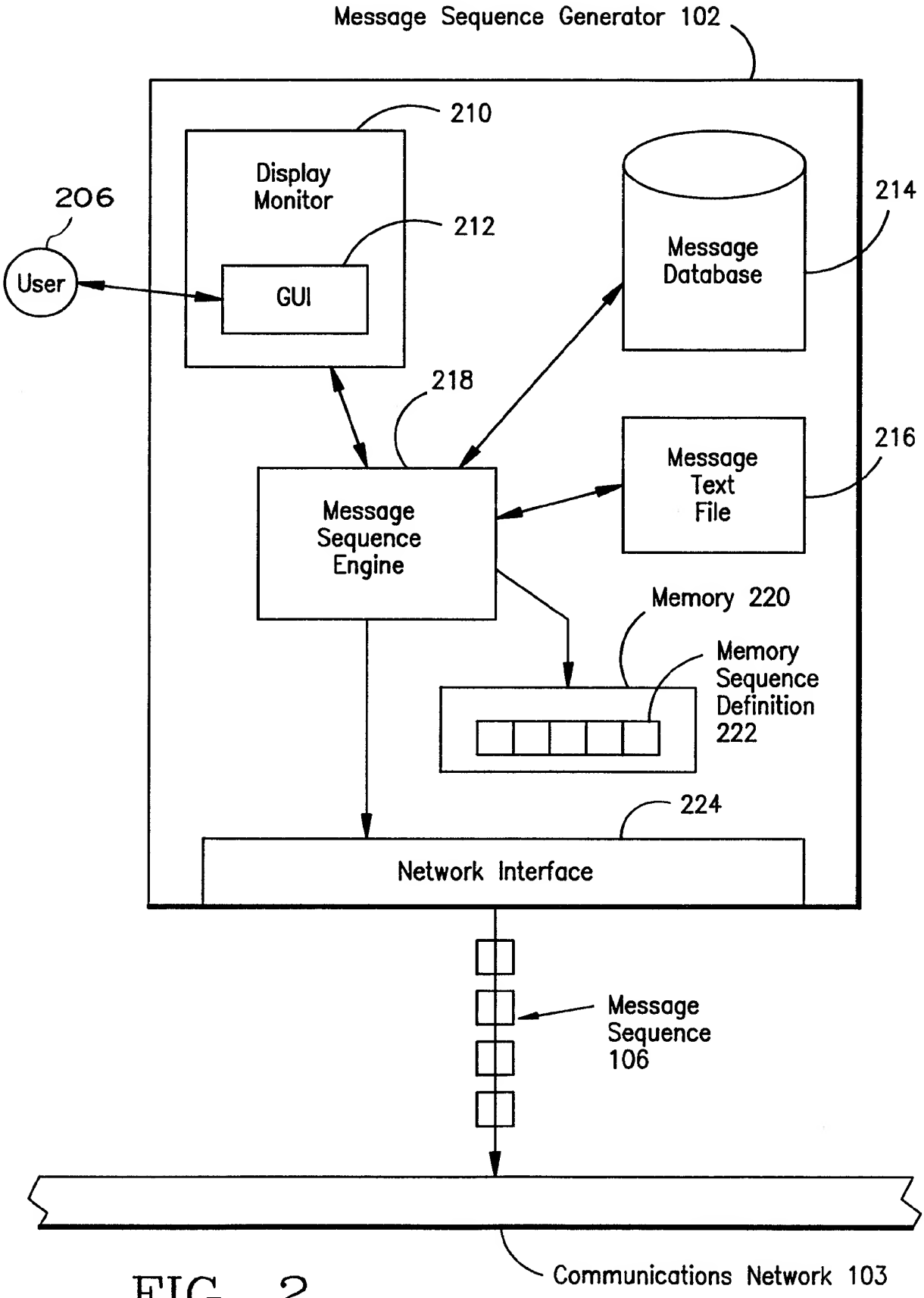

Message Sequence Generator 102
210
Display Monitor
212
GUI
206
User
214
Message Database
218
216
Message Text File
Message Sequence Engine
Memory 220
Memory Sequence Definition 222
224
Network Interface
Message Sequence 106
Communications Network 103

FIG. 2

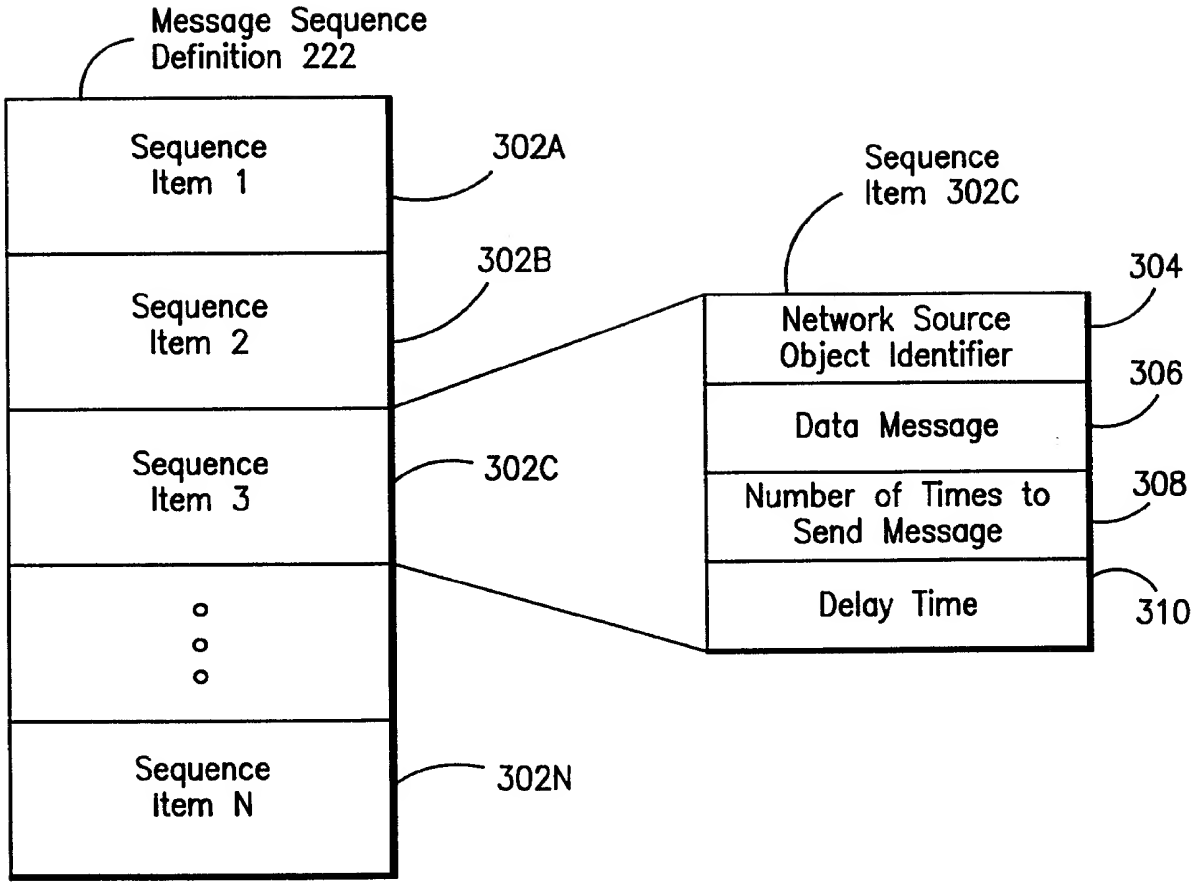

Message Sequence Definition 222
Sequence Item 1
Sequence Item 2
Sequence Item 3
o
o
o
Sequence Item N
302A
302B
302C
302N
Sequence Item 302C
Network Source Object Identifier
Data Message
Number of Times to Send Message
Delay Time
304
306
308
310

FIG. 3

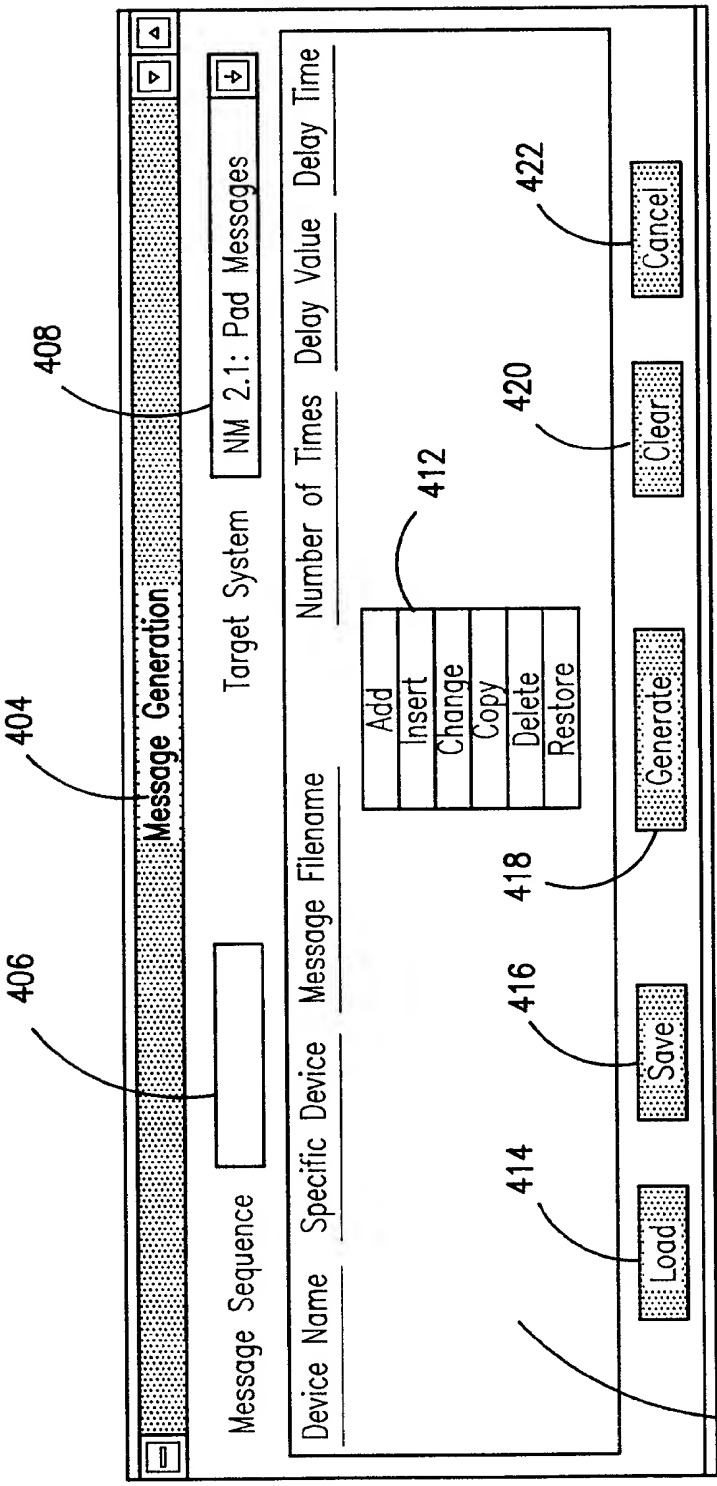

Message Generation
Message Sequence
Target System
NM 2.1: Pad Messages
Device Name
Specific Device
Message Filename
Number of Times
Delay Value
Delay Time
Add
Insert
Change
Copy
Delete
Restore
Load
Save
Generate
Clear
Cancel
404
406
408
410
412
414
416
418
420
422
Message Sequence Display Window 410

FIG. 4A

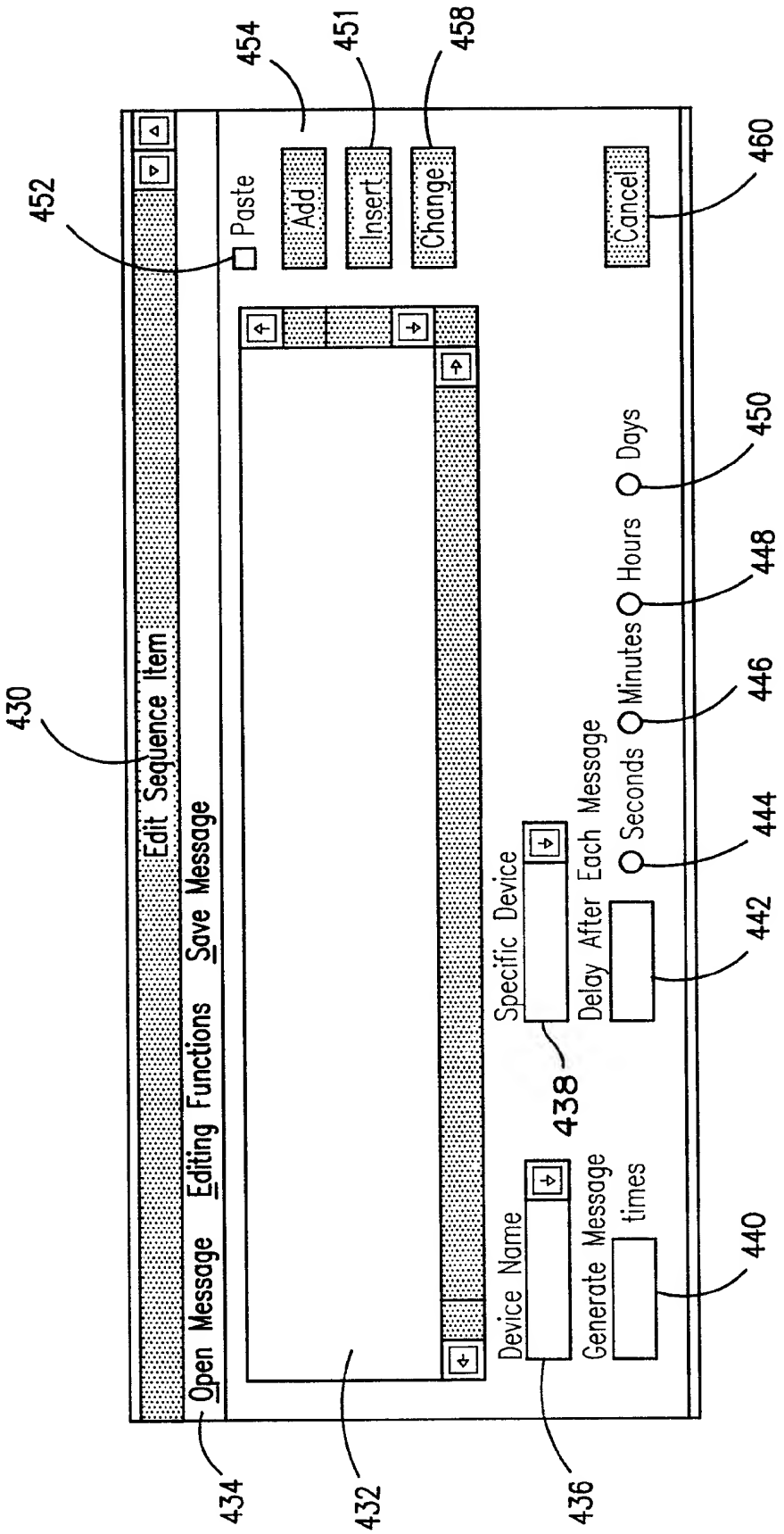

454
451
458
452
460
Paste
Add
Insert
Change
Cancel
450
Days
448
Hours
446
Minutes
444
Seconds
Delay After Each Message
442
Specific Device
438
Device Name
Generate Message
times
440
430
Edit Sequence Item
Open Message
Editing Functions
Save Message
434
432
436

FIG. 4B

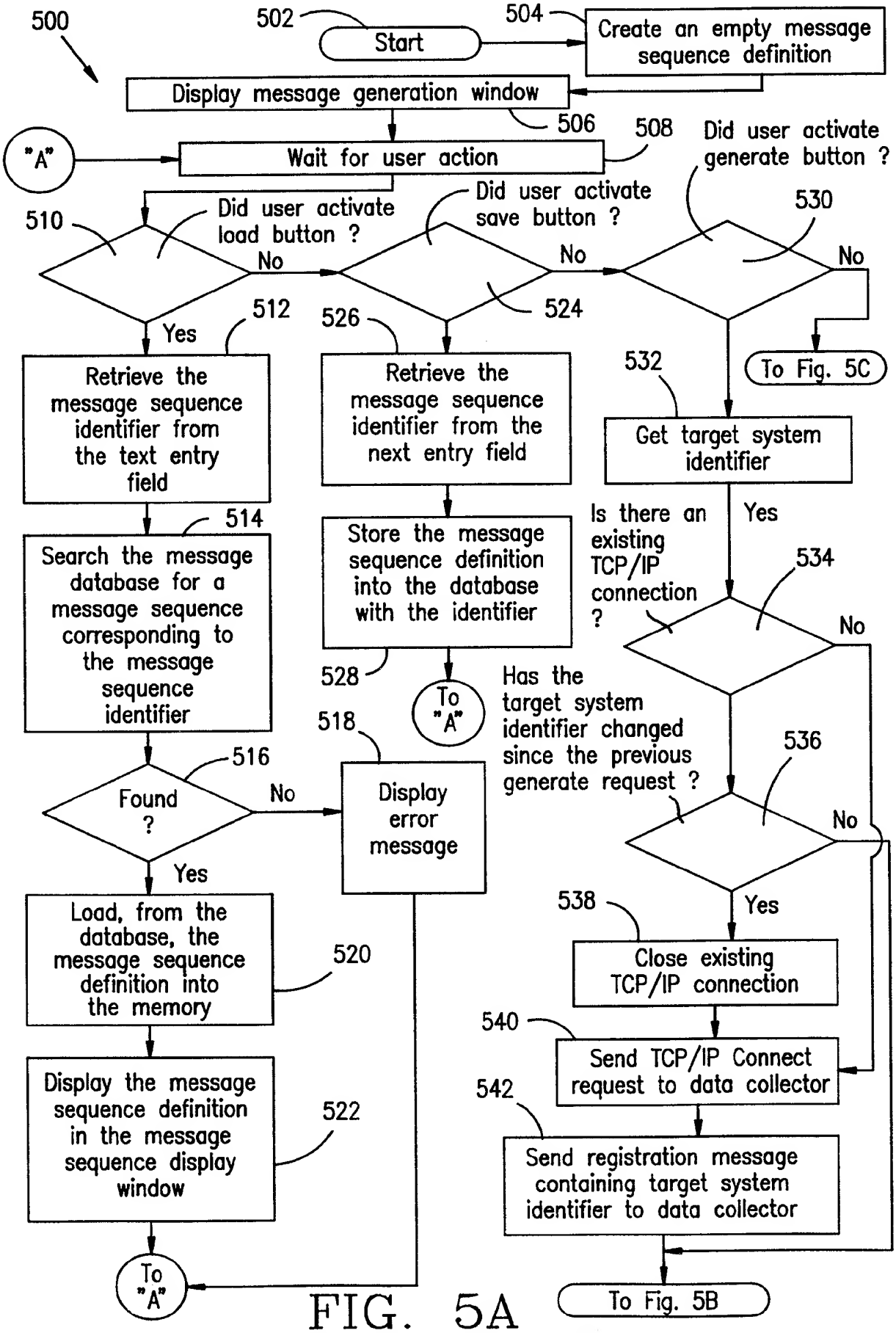

500
502
Start
504
Create an empty message sequence definition
Display message generation window
506
"A"
Wait for user action
508
Did user activate generate button ?
510
Did user activate load button ?
No
Did user activate save button ?
No
530
No
Yes
512
526
524
Retrieve the message sequence identifier from the text entry field
Retrieve the message sequence identifier from the next entry field
To Fig. 5C
532
Get target system identifier
514
Search the message database for a message sequence corresponding to the message sequence identifier
Store the message sequence definition into the database with the identifier
Is there an existing TCP/IP connection ?
Yes
534
No
528
To "A"
Has the target system identifier changed since the previous generate request ?
518
516
Found ?
No
Display error message
536
No
Yes
538
Load, from the database, the message sequence definition into the memory
520
Close existing TCP/IP connection
540
Send TCP/IP Connect request to data collector
Display the message sequence definition in the message sequence display window
522
542
Send registration message containing target system identifier to data collector
To "A"
To Fig. 5B

FIG. 5A

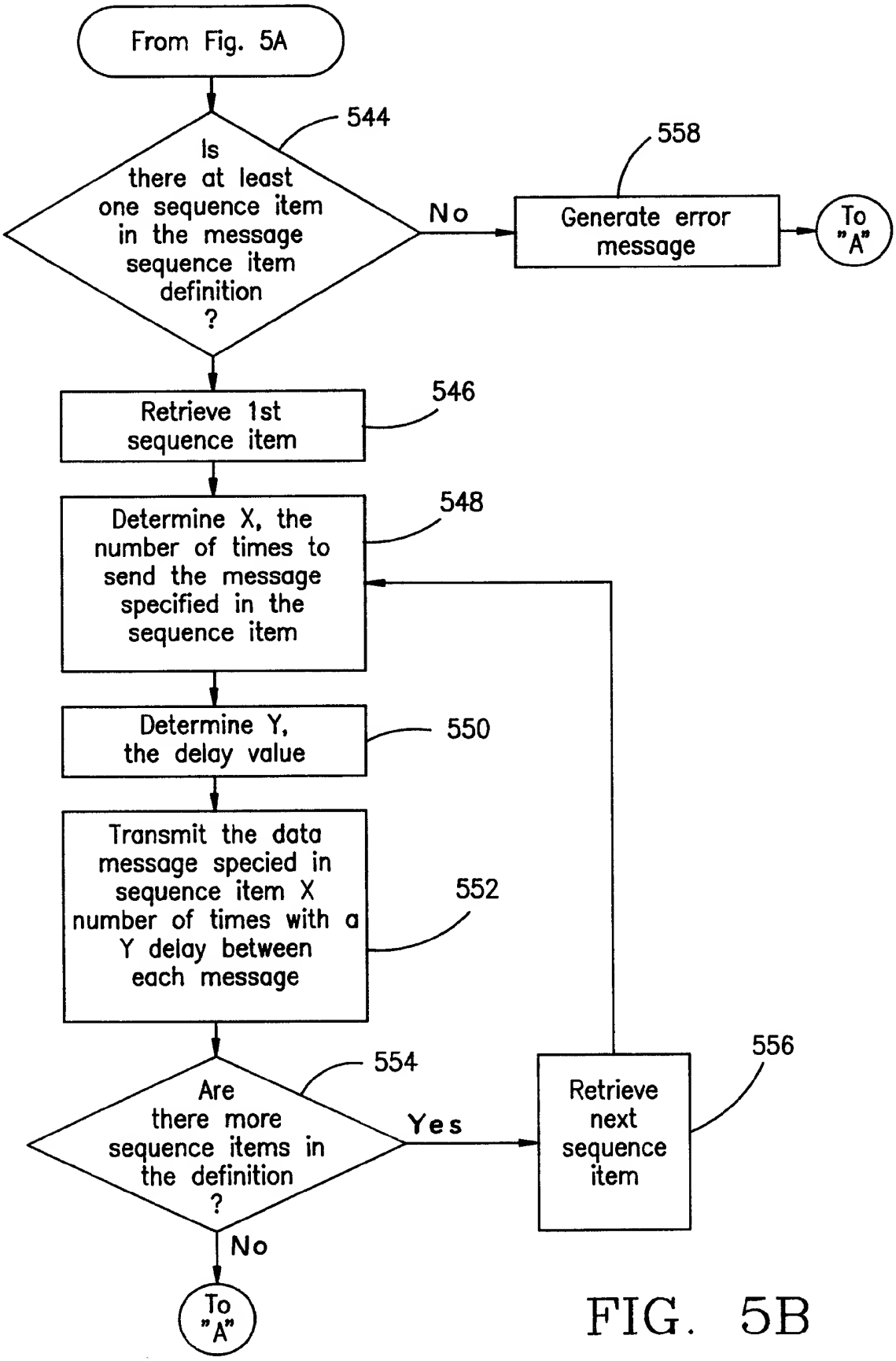

From Fig. 5A
544
Is there at least one sequence item in the message sequence item definition ?
No
558
Generate error message
To "A"
546
Retrieve 1st sequence item
548
Determine X, the number of times to send the message specified in the sequence item
550
Determine Y, the delay value
552
Transmit the data message specied in sequence item X number of times with a Y delay between each message
554
Are there more sequence items in the definition ?
Yes
556
Retrieve next sequence item
No
To "A"

FIG. 5B

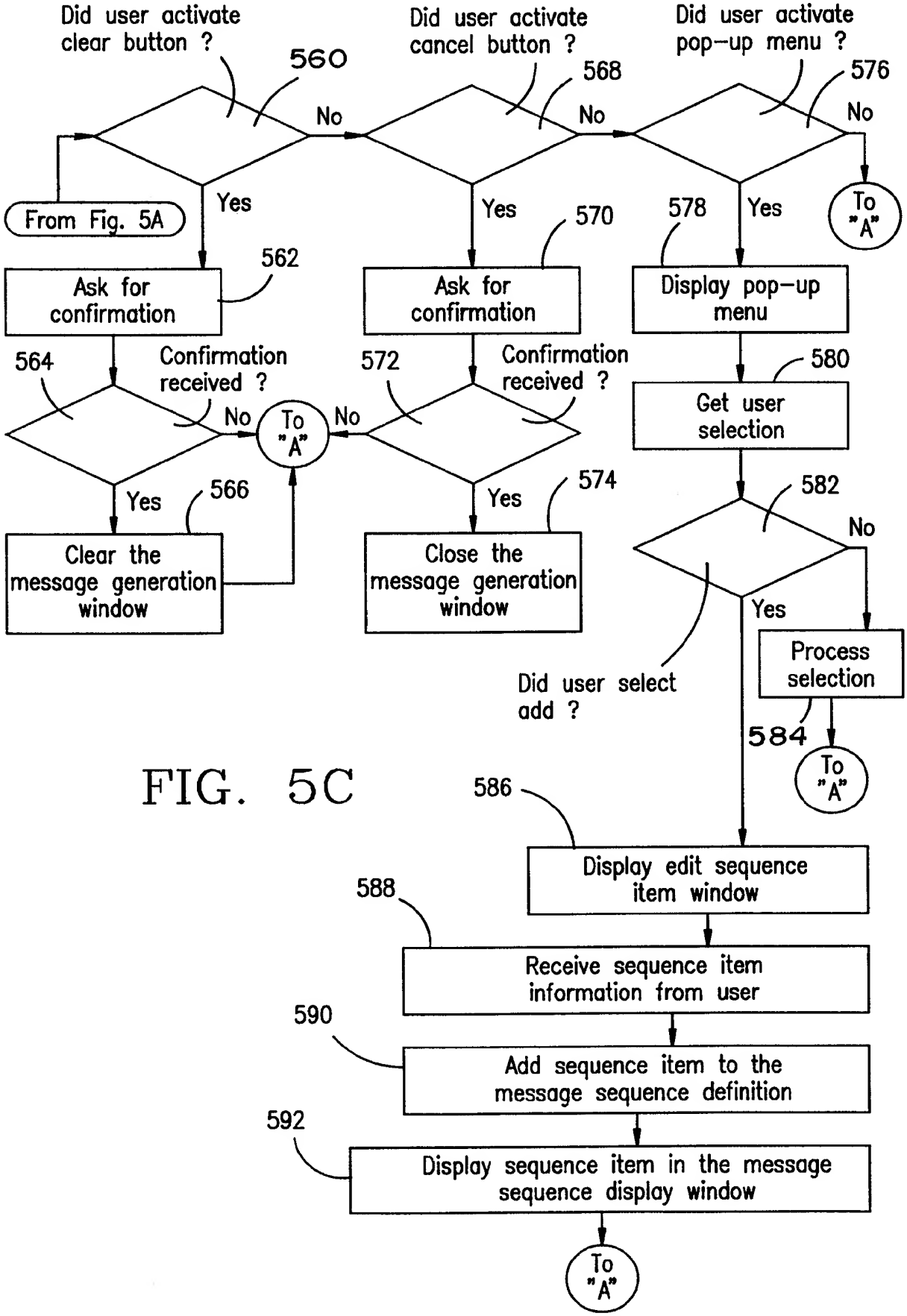

Did user activate clear button ?
560
No
Yes
From Fig. 5A
562
Ask for confirmation
564
Confirmation received ?
No
To "A"
Yes
566
Clear the message generation window
Did user activate cancel button ?
568
No
Yes
570
Ask for confirmation
572
Confirmation received ?
No
Yes
574
Close the message generation window
Did user activate pop-up menu ?
576
No
To "A"
578
Yes
Display pop-up menu
580
Get user selection
582
No
Yes
Process selection
584
To "A"
Did user select add ?
586
Display edit sequence item window
588
Receive sequence item information from user
590
Add sequence item to the message sequence definition
592
Display sequence item in the message sequence display window
To "A"

FIG. 5C

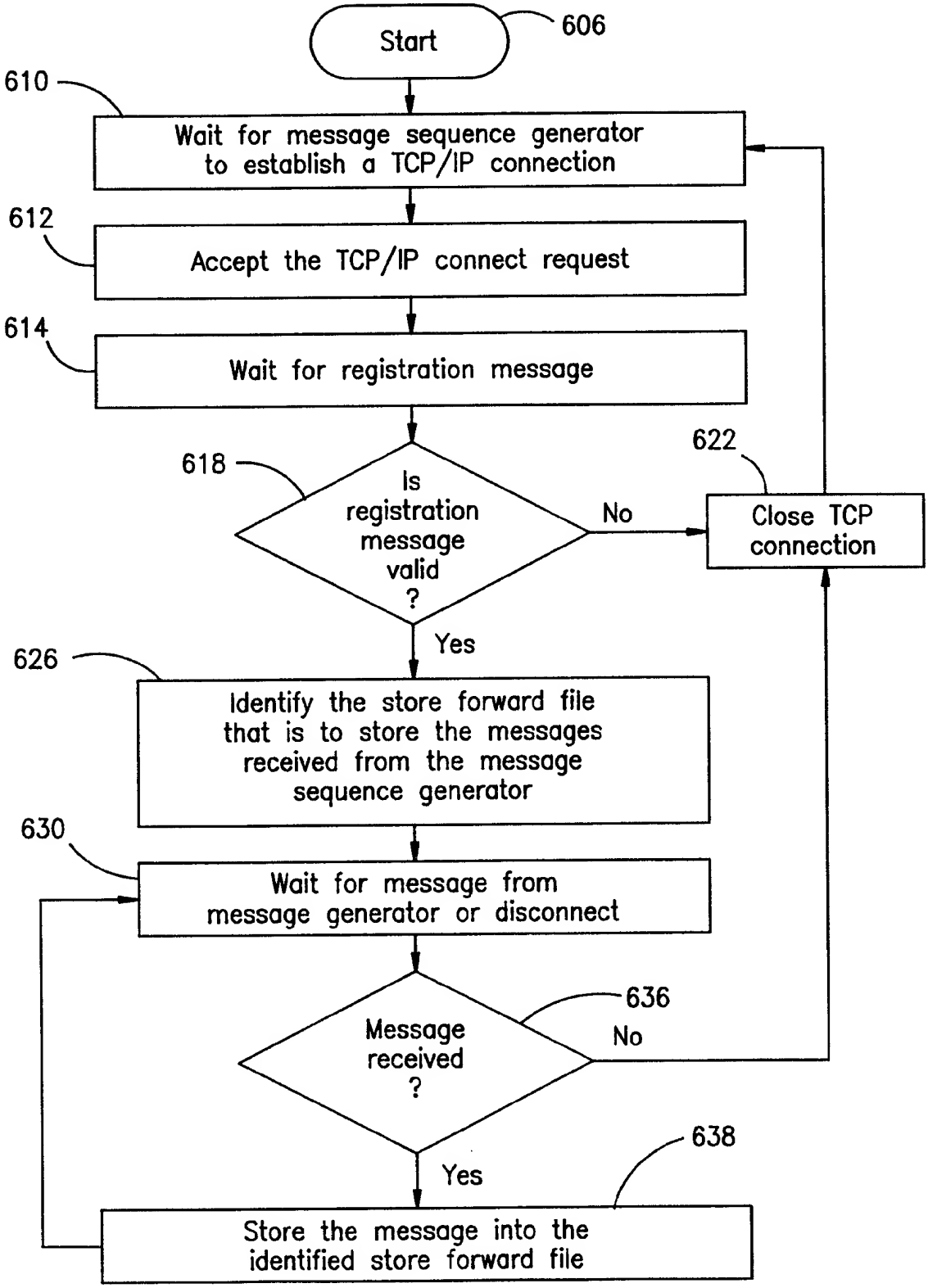

Start
606
610
Wait for message sequence generator to establish a TCP/IP connection
612
Accept the TCP/IP connect request
614
Wait for registration message
618
Is registration message valid ?
No
622
Close TCP connection
Yes
626
Identify the store forward file that is to store the messages received from the message sequence generator
630
Wait for message from message generator or disconnect
636
Message received ?
No
Yes
638
Store the message into the identified store forward file

FIG. 6

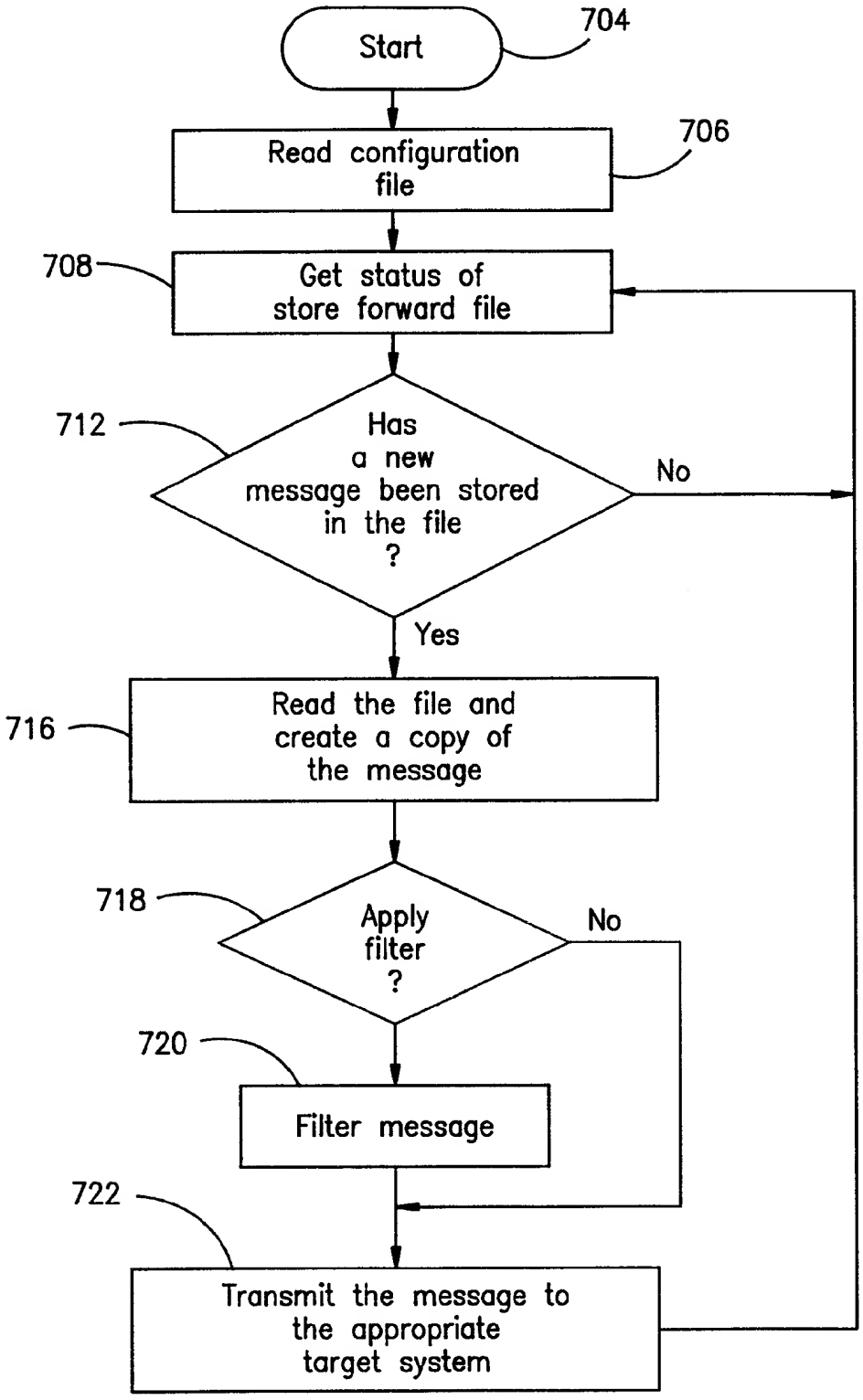

Start
704
Read configuration file
706
708
Get status of store forward file
712
Has a new message been stored in the file ?
No
Yes
716
Read the file and create a copy of the message
718
Apply filter ?
No
720
Filter message
722
Transmit the message to the appropriate target system

FIG. 7

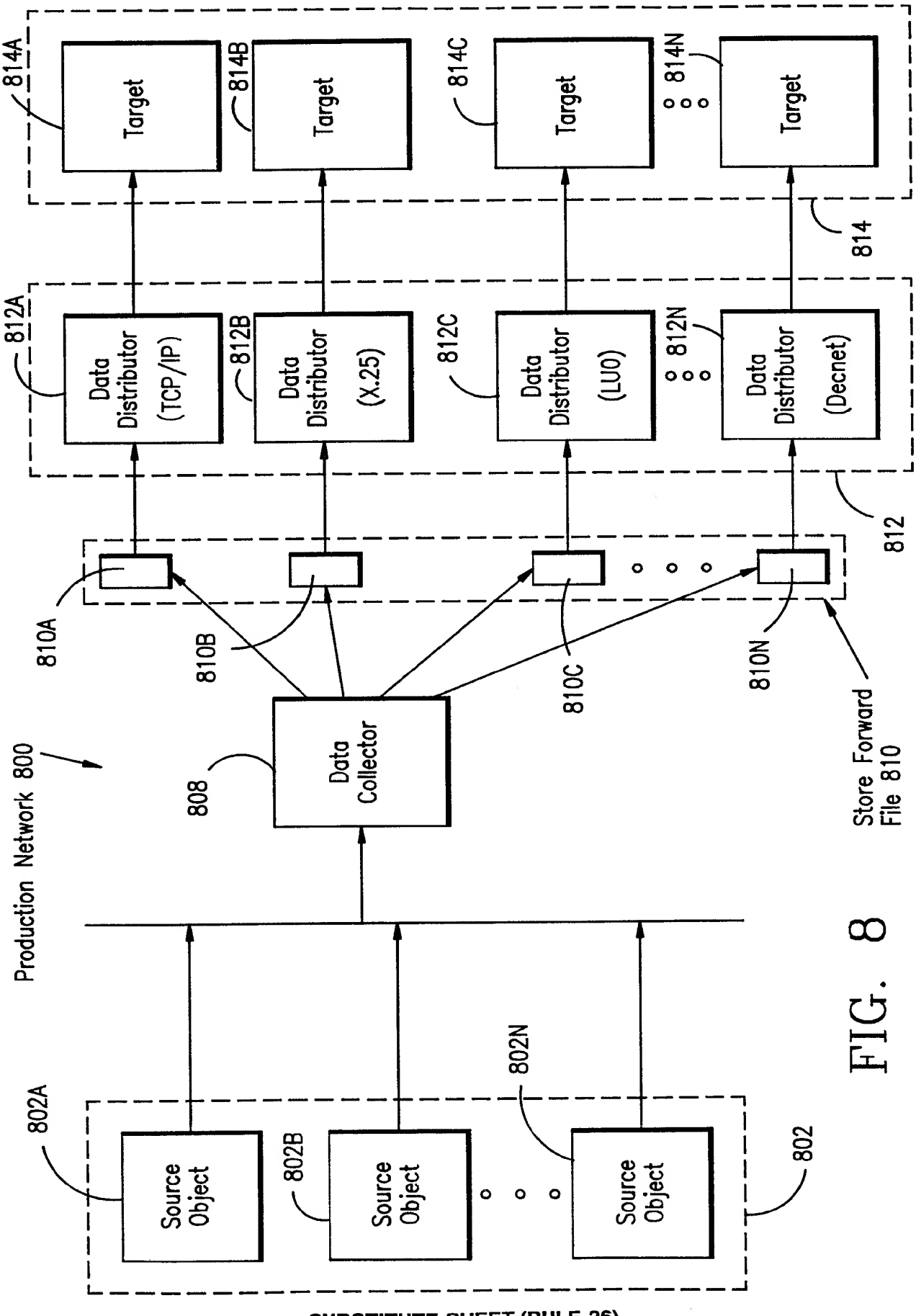

Production Network 800
802A
Source Object
802B
Source Object
802N
Source Object
802
808
Data Collector
810A
810B
810C
810N
Store Forward File 810
812A
Data Distributor (TCP/IP)
812B
Data Distributor (X.25)
812C
Data Distributor (LU0)
812N
Data Distributor (Decnet)
812
814A
Target
814B
Target
814C
Target
814N
Target
814

FIG. 8

FIG. 9

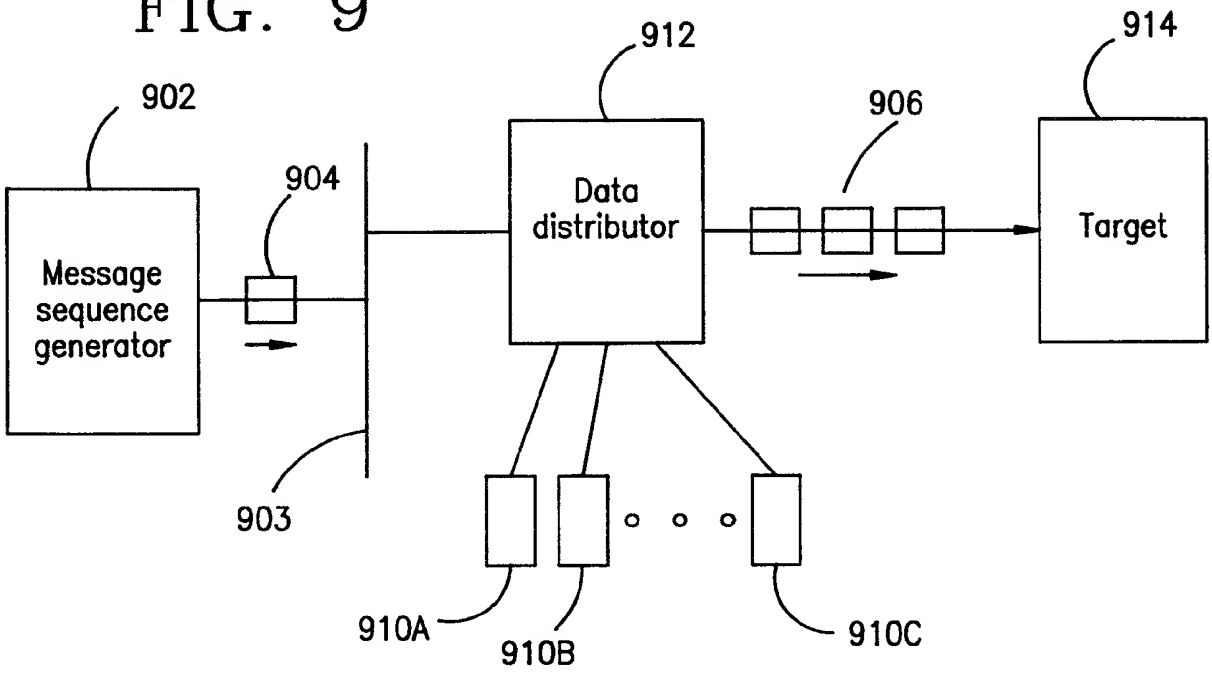
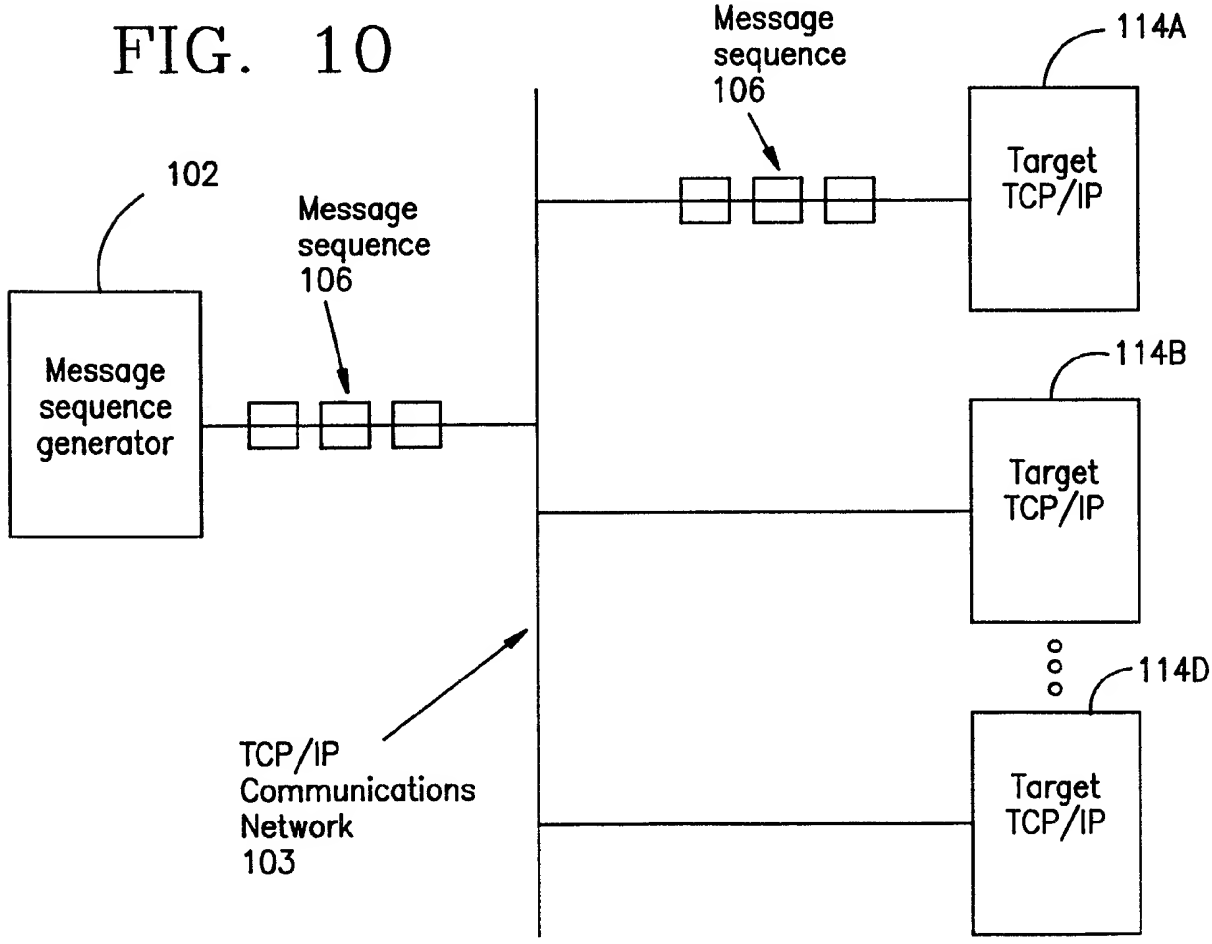

902
904
912
906
914
Message sequence generator
Data distributor
Target
903
910A
910B
910C

FIG. 10


Message sequence 106
114A
102
Message sequence 106
Target TCP/IP
Message sequence generator
114B
Target TCP/IP
114D
TCP/IP Communications Network 103
Target TCP/IP

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US98/11801

A. CLASSIFICATION OF SUBJECT MATTER

IPC(6) :G06F 15/16, 13/42
US CL :395/200.54, 183.22

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

U.S. : 395/200.54, 183.22

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

APS

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	US 5,063,523 A (Vrenjak) 05 November 1991 (05.11.91), abstract, fig.1.	1-18
Y	US 5,627,766 A (Beaven) 06 May 1997 (06.05.97), col.1-3.	1-18
Y	US 5,586,255 A (Tanaka et al.) 17 December 1996 (07.12.96), abstract.	1-18
Y, P	US 5,751,962 A (Fanshier et al.) 12 May 1998 (12.05.98), abstract.	1-18

☐ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

*	Special categories of cited documents:	"T"	later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"A"	document defining the general state of the art which is not considered to be of particular relevance	"X"	document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"E"	earlier document published on or after the international filing date	"Y"	document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"L"	document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)	"&"	document member of the same patent family
"O"	document referring to an oral disclosure, use, exhibition or other means		
"P"	document published prior to the international filing date but later than the priority date claimed		

Date of the actual completion of the international search 08 SEPTEMBER 1998	Date of mailing of the international search report 13 OCT 1998
Name and mailing address of the ISA/US Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231 Facsimile No. (703) 305-3230	Authorized officer DUNG DINH Telephone No. (703) 305-9600

Form PCT/ISA/210 (second sheet)(July 1992)★